REMARKS

Claims 1, 4-8, 11-15, 18-19, 22-28, 30-32, 34-36 and 38-39 are pending in the present application. Applicants respectfully request reconsideration of the application in view of the above amendments and remarks made herein.

I. Rejections Under 35 U.S.C. § 101

Claims 1, 3-8, 11-14 and 38-39 are rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. The Examiner essentially stated that the claims, at most, incorporate ineligible transformations, and are thus not tied to a statutory class under 35 U.S.C. § 101.

Claims 1 and 38 are the independent claims.

Applicants appreciate the Examiner's suggestion to tie the claims to a valid statutory class (i.e., a computer-implemented method).

Claims 1 and 38 claim, *inter alia*, "a computer-implemented method...parsing a logical structure of the declared object from the entity definition by a processor..." (emphasis added).

Claims 1 and 38 are believed to be directed to statutory subject matter under 35 U.S.C. § 101, as both claims are tied to a computer-implemented method utilizing a processor and a persistent medium.

Applicants respectfully request that inasmuch as Claims 4-8, 11-14 and 39 are dependent on Claims 1 and 38, and Claims 1 and 38 are patentable, Claims 4-8, 11-14 and 39 are patentable as dependent on patentable independent claims. Claim 3 is canceled. Reconsideration of the instant rejection is respectfully requested.

Claims 15, 17-19, 22-32 and 34-36 are rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. The Examiner stated essentially that all of the claims are medium or system claims comprising only module or other logic based limitations, and without any accompanying hardware to realize the functionality of the systems, the claims do not fall within one of the statutory categories of invention as defined in 35 U.S.C. § 101.

Claims 15 and 26 are the independent claims.

Claim 15 claims, *inter alia*, a "computer-readable medium embodying instructions executed by a processor to perform method steps for generating a persistent storage system." Claim 15 further claims, *inter alia*, a combination of a persistent storage structure, an interface, a database table, and an index. Claim 26 claims, *inter alia*, a "persistent storage system." Claim 26 further claims, *inter alia*, a combination of an interface, a utility module, a persistent storage structure, a persistence module, a storage mapping module, a database table, an access interface, and an index creation module.

Claims 15 and 26 are believed to be directed to statutory subject matter. Claim 15 is a Beauregard type claim and Claim 26 is a product claim. Consider the following from *Ex parte Bo Li*, Appeal 2008-1213, decided November 6, 2008:

> "In the analysis of In re Nuijten, 500 F.3d 1346 (Fed. Cir., 2007), the Federal Circuit considers the four statutory classes for a signal, and bases the determination of statutory subject matter on that basis. It has been the practice for a number of years that a "Beauregard Claim" of this nature be considered statutory at the USPTO as a product claim (MPEP 2105.01, I). Though not finally adjudicated, this practice is not inconsistent with In re Nuijten. Further,

> the instant claim presents a number of software components, such as the claimed logic processing module, configuration file processing module, data organization module, and data display organization module, that are embodied upon a computer readable medium. This combination has been found statutory under the teachings of In re Lowry, 32 F.3d 1579 (Fed. Cir. 1994)."

In view of the foregoing, Claim 15 is believed to be directed to statutory subject matter under 35 U.S.C. § 101. Further, similar to the combination of software components in *Bo Li*, the combinations claimed in Claims 15 and 26 are believed to be directed to statutory subject matter under 35 U.S.C. § 101.

Applicants respectfully request that inasmuch as Claims 18-19, 22-25, 27-28, 30-32 and 34-36 are dependent on Claims 15 and 26, and Claims 15 and 26 are patentable, Claims 18-19, 22-25, 27-28, 30-32 and 34-36 are patentable as dependent on patentable independent claims. Claims 17 and 29 are canceled. Reconsideration of the instant rejection is respectfully requested.

II. Rejections Under 35 U.S.C. § 103

Claims 1, 3-8, 11-15, 17-19, 22-32, 34-36 and 38-39 are rejected under 35 U.S.C. § 103(a) as being unpatentable over *Bannon et al.* (US 5,297,279) in view of *Collins* (US 7,028,025), in further view of *Murthy et al.* (US 2003/0140308). The Examiner essentially stated that the combined teachings of *Bannon*, *Collins* and *Murthy* teach or suggest all of the limitations of Claims 1, 3-8, 11-15, 17-19, 22-32, 34-36 and 38-39.

Claims 1, 15, 26 and 38 are the independent claims.

Claims 1 and 38 claim, *inter alia*, "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically to enable management of object instance data in the persistent storage structure." Claim 15 claims, *inter alia*, "generating automatically an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically to enable management of object instance data in the persistent storage structure." Claim 26 claims, *inter alia*, "wherein the autonomous persistent storage structure is automatically configured to comprise: an access interface comprising access object classes that are generated automatically to enable management of the object instance data in the persistent storage structure."

Bannon teaches an object-oriented database (OODB) for providing long-term storage and retrieval of objects created by application programs written at least in part in object-oriented programming languages (see col. 5, lines 39-45). *Bannon* does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure" (emphasis added) as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. *Bannon* teaches persistent object references (PTR instances) to create, manipulate, store, retrieve, and access persistent objects in the OODB (see col. 9, lines 52-56 and col. 21, lines 55-61). That is, *Bannon* teaches PTR instances to enable management of object instance data in the OODB. As FIG. 2 of *Bannon* illustrates, PTR instances (44) are not located within the OODB (18); rather, they are separate and apart. Thus, *Bannon* does not teach or suggest managing object instance data via an

interface located within the persistent storage structure, essentially as claimed in Claims 1, 15, 26 and 38. Compare FIG. 2 of *Bannon* with FIG. 4 of the Application:

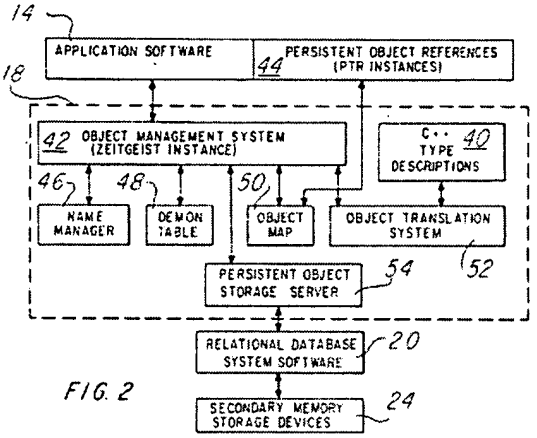

FIG. 2

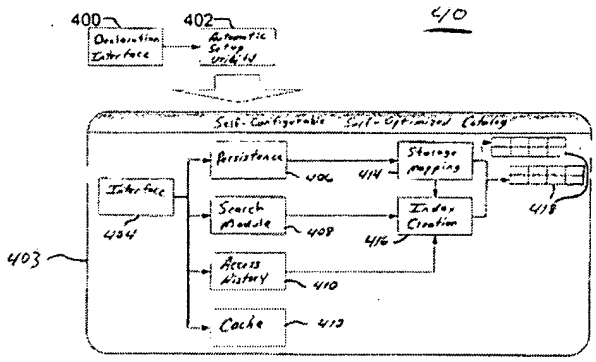

Fig. 4

As shown in FIG. 2, *Bannon* requires an external component (PTR instances (44)) to be interfaced to the OODB (18) in order to manage the object instance data in the OODB. By way of comparison, the interface (404) which manages the object instance data in the persistent storage structure (403) in Claims 1, 15, 26 and 38, is located within the persistent storage structure (403) (see FIG. 4 of the Application). Utilizing a separate, external component located outside of the OODB to manage the object instance data in the OODB, as taught by *Bannon*, is not analogous to an interface within the persistent storage structure enabling management of object instance data within the persistent storage structure. Thus, *Bannon* fails to teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Collins teaches improving the performance of distributed systems by reducing the amount of graphical data transmitted between a server and a client (see col. 1, line 65 – col. 2, line 2). *Collins* teaches transmitting indicià of a bitmap rather than the relatively larger, encoded

bitmap, to improve the encoding and compression of graphical data that has been previously encountered during a particular client-server session (see col. 12, lines 50-60). *Collins* does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure" as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. Indeed, *Collins* is not applied in the rejection for this teaching. Therefore, *Collins* fails to cure the deficiencies of *Bannon*.

Murthy teaches a method allowing users to register XML schemas in a database system and mapping constructs defined in the XML schema to constructs supported by the database system (see paragraph [0032]). *Murthy* does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent storage structure" as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. Indeed, *Murthy* is not applied in the rejection for this teaching. Therefore, *Murthy* fails to cure the deficiencies of *Bannon* and *Collins*.

The combination of *Bannon*, *Collins* and *Murthy* teaches an object-oriented database for providing long-term storage and retrieval of objects with improved performance by reducing the amount of any graphical data transmitted between a server and a client and mapping constructs defined in an XML schema to constructs supported by the database. The combination does not teach or suggest "generating automatically, by the processor, an interface within the persistent storage structure, wherein the interface comprises access object classes that are generated automatically by the processor to enable management of object instance data in the persistent

storage structure" as claimed in Claim 1 and essentially as claimed in Claims 15, 26 and 38. Accordingly, the combination does not teach or suggest every limitation of Claims 1, 15, 26 and 38.

Claims 1 and 38 further claim, *inter alia*, "generating automatically, by the processor, a database table within the persistent storage structure to store the object instance data." Claim 15 further claims, *inter alia*, "generating automatically a database table within the persistent storage structure to store the object instance data." Claim 26 further claims, *inter alia*, "wherein the persistent storage structure is automatically configured to comprise: a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data."

Bannon teaches an object-oriented database (OODB) for providing long-term storage and retrieval of objects created by application programs written at least in part in object-oriented programming languages (see col. 5, lines 39-45). *Bannon* does not teach or suggest generating automatically "a database table within the persistent storage structure to store the object instance data" as claimed in Claims 1, 15 and 38, or a persistent storage structure comprising "a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data" as claimed in Claim 26. For example, compare FIG. 2 of *Bannon with* FIG. 4 of the Application:

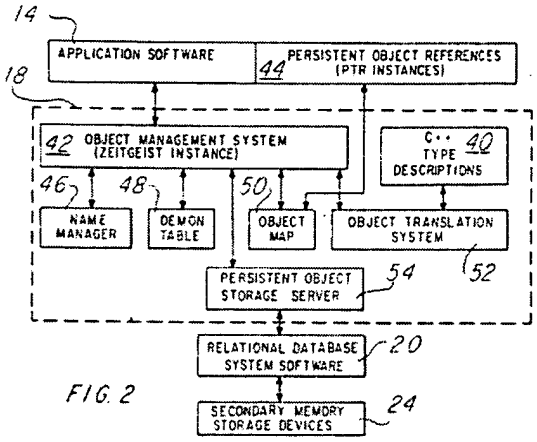

FIG. 2

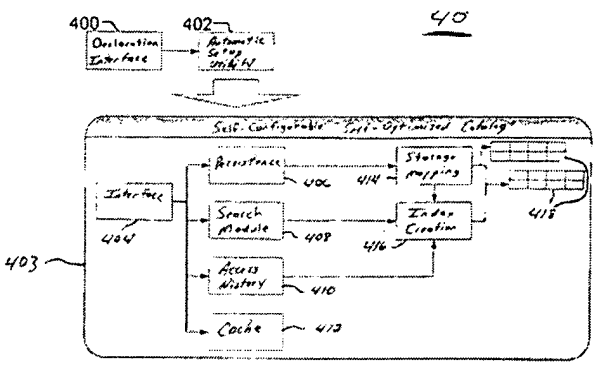

Fig. 4

Bannon teaches storing persistent objects in a persistent object server (POS server) (54) that is located within the OODB (18) (see FIG. 2 and col. 10, lines 11-15). *Bannon* teaches controlling this POS server (54) with relational database system software (RDBMS) (20) that is located outside of the OODB (see FIG. 2 and col. 9, lines 28-37). By way of comparison, the database table and all corresponding software to control the database table in Claims 1, 15, 26 and 38 are contained within the persistent storage structure (see FIG. 4 of the Application). Thus, the two are clearly not analogous. Further, referring specifically to Claim 26, Claim 26 claims a persistent storage structure (403) wherein the database table (418) is automatically generated by a persistence module (406) and a storage mapping module (414) (see FIG. 4 of the Application). Nowhere does *Bannon* teach or suggest a persistence module or a storage mapping module. Further, assuming, *arguendo*, that the RDBMS (20) of *Bannon* is analogous to the persistence module and storage mapping module in Claim 26, the RDBMS (20) of *Bannon* is clearly not located within the OODB (18) (see FIG. 2 of *Bannon*), as stated above. Thus, *Bannon* does not teach or suggest all of the limitations of Claims 1, 15, 26 and 38.

Collins teaches improving the performance of distributed systems by reducing the amount of graphical data transmitted between a server and a client (see col. 1, line 65 – col. 2, line 2). *Collins* teaches transmitting indicia of a bitmap rather than the relatively larger, encoded bitmap, to improve the encoding and compression of graphical data that has been previously encountered during a particular client-server session (see col. 12, lines 50-60). *Collins* does not teach or suggest generating automatically "a database table within the persistent storage structure to store the object instance data" as claimed in Claims 1, 15 and 38, or a persistent storage structure comprising "a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data" as claimed in Claim 26. Indeed, *Collins* is not applied in the rejection for this teaching. Therefore, *Collins* fails to cure the deficiencies of *Bannon*.

Murthy teaches a method allowing users to register XML schemas in a database system and mapping constructs defined in the XML schema to constructs supported by the database system (see paragraph [0032]). *Murthy* does not teach or suggest generating automatically "a database table within the persistent storage structure to store the object instance data" as claimed in Claims 1, 15 and 38, or a persistent storage structure comprising "a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data" as claimed in Claim 26. Indeed, *Murthy* is not applied in the rejection for this teaching. Therefore, *Murthy* fails to cure the deficiencies of *Bannon* and *Collins*.

The combination of *Bannon*, *Collins* and *Murthy* teaches an object-oriented database for providing long-term storage and retrieval of objects with improved performance by reducing the amount of any graphical data transmitted between a server and a client and mapping constructs

defined in an XML schema to constructs supported by the database. The combination does not teach or suggest generating automatically "a database table within the persistent storage structure to store the object instance data" as claimed in Claims 1, 15 and 38, or a persistent storage structure comprising "a persistence module and a storage mapping module for automatically generating a database table within the persistent storage structure for storing object instance data" as claimed in Claim 26. Accordingly, the combination does not teach or suggest every limitation of Claims 1, 15, 26 and 38.

Claims 1 and 15 further claim, *inter alia*, generating automatically "an index to object instance data if it is determined that a frequency of accessing the object instance data exceeds a predefined threshold."

Bannon teaches an object-oriented database (OODB) for providing long-term storage and retrieval of objects created by application programs written at least in part in object-oriented programming languages (see col. 5, lines 39-45). The Examiner concedes that *Bannon* does not teach or suggest generating automatically "an index to object instance data if it is determined that a frequency of accessing the object instance data exceeds a predefined threshold" (see page 5 of the Office Action). Therefore, *Bannon* fails to teach or suggest all of the limitations of Claims 1 and 15.

Collins teaches improving the performance of distributed systems by reducing the amount of graphical data transmitted between a server and a client (see col. 1, line 65 – col. 2, line 2). *Collins* does not teach or suggest generating automatically "an index to object instance data if it is determined that a frequency of accessing the object instance data exceeds a predefined threshold" (emphasis added) as claimed in Claims 1 and 15. Consider that in order to

reduce the amount of graphical data transmitted between the server and the client, *Collins* teaches storing the indicia of a bitmap rather than the relatively larger, encoded bitmap itself. This is done for every bitmap the first time the bitmap is encountered during a client-server session, and the complete index that results from storing every bitmap encountered is maintained during that client-server session (see col. 12, lines 48-60). That is, *Collins* does not teach or suggest generating an index to object instance data if the frequency of accessing the object instance data exceeds a predefined threshold, essentially as claimed in Claims 1 and 15.

Indeed, *Collins* arguably teaches away from selectively indexing bitmaps based on a predefined threshold. Consider that the object of the invention of *Collins* is to reduce the amount of graphical data transmitted between a server and a client (see Abstract and col. 1, line 65 – col. 2, line 2). With this in mind, a person having ordinary skill in the art would likely be motivated to index every bitmap transferred during a client-server session, at the expense of maintaining a potentially large index, in order to minimize the amount of graphical data transmitted between the server and the client during that client-server session; indeed, this is exactly the approach taken in *Collins*. Selectively indexing only object instance data that exceeds a predefined threshold, essentially as claimed in Claims 1 and 15, is counterproductive to reducing the amount of graphical data transmitted between a server and a client – the explicit goal of *Collins*.

For at least the foregoing reasons, *Collins* fails to cure the deficiencies of *Bannon*.

Murthy teaches a method allowing users to register XML schemas in a database system and mapping constructs defined in the XML schema to constructs supported by the database system (see paragraph [0032]). *Murthy* does not teach or suggest generating automatically "an index to object instance data if it is determined that a frequency of accessing the object instance data exceeds a predefined threshold" as claimed in Claims 1 and 15. Indeed, *Murthy* is not

applied in the rejection for this teaching. Therefore, *Murthy* fails to cure the deficiencies of *Bannon* and *Collins*.

The combination of *Bannon*, *Collins* and *Murthy* teaches an object-oriented database for providing long-term storage and retrieval of objects with improved performance by reducing the amount of any graphical data transmitted between a server and a client and mapping constructs defined in an XML schema to constructs supported by the database. The combination does not teach or suggest generating automatically "an index to object instance data if it is determined that a frequency of accessing the object instance data exceeds a predefined threshold" as claimed in Claims 1 and 15. Accordingly, the combination does not teach or suggest every limitation of Claims 1 and 15.

Therefore, for at least the reasons above, Claims 1, 15, 26 and 38 are believed to be patentable and non-obvious over the combination of *Bannon*, *Collins* and *Murthy*. Applicants respectfully submit that inasmuch as Claims 4-8, 11-14, 18-19, 22-25, 27-28, 30-32, 34-36 and 39 are dependent on Claims 1, 15, 26 and 38, and Claims 1, 15, 26 and 38 are patentable over the cited references, Claims 4-8, 11-14, 18-19, 22-25, 27-28, 30-32, 34-36 and 39 are patentable as dependent on patentable independent claims. Claims 3, 17 and 29 are canceled. Reconsideration of the instant rejection is respectfully requested.

CONCLUSION

In view of the foregoing, it is believed that all claims now pending patentability define the subject invention over the prior art of record and are in condition for allowance.

Early and favorable reconsideration of the case is respectfully requested.

Respectfully submitted,

Date: June 11, 2009

By: /Nathaniel T. Wallace/
Nathaniel T. Wallace
Reg. No. 48,909
Attorney for Applicant(s)

F. Chau & Associates, LLC
130 Woodbury Road
Woodbury, New York 11797
TEL: (516) 692-8888
FAX: (516) 692-8889